Amendments to the Claims

1. (Currently Amended) A computer-implemented method for passing a message from a first thread of execution in a process to a second thread of execution in the process, ~~the first thread being adapted to interpret a block of source code, the second thread having a queue for holding messages, the method~~ comprising:

instantiating a queue for holding messages at the second thread of execution in the process, each queue comprising a reference to a further queue of the same type;

interpreting a block of source code at the first thread of execution in the process;

instantiating the message at the first thread of execution in the process;

placing, by the first thread, a reference to the message into the queue of the second thread, wherein the reference is usable by the second thread to access the message.

2. (Original) A computer-readable medium having stored thereon computer-executable instructions for performing the method of claim 1.

3. (Previously Presented) The method of claim 1 further comprising:

receiving a reference to the second thread's queue; and

using the reference to the second thread's queue to perform the placing step.

4. (Previously Presented) The method of claim 1, wherein the first thread has a queue, the method further comprising: passing, to the second thread, a reference to the first thread's queue to allow the second thread to send messages to the first thread.

5. (Previously Presented) The method of claim 1 further comprising: sending a signal to the second thread to indicate that a message has been sent to the second thread.

6. (Original) The method of claim 5, wherein the signal is sent via a platform-independent object.

7. (Previously Presented) The method of claim 1 further comprising:

defining a message object for holding the message; and

inserting the message into the message object, wherein the reference placed in the second thread's queue is a reference to the message object.

8. (Currently Amended) A method for passing intraprocess messages between scripting threads in a process, the method comprising:

creating a first scripting thread of execution;

creating a first queue for the first scripting thread, the first queue comprising a reference to a second queue of the same type as the first queue;

creating a second scripting thread of execution; and

passing, to the second scripting thread, a reference to the first scripting thread's queue for use by the second scripting thread to send messages to the first scripting thread.

9. (Original) A computer-readable medium having stored thereon computer-executable instructions for performing the method of claim 8.

10. (Previously Presented) The method of claim 8 further comprising:

creating a queue for the second scripting thread; and

passing, to the first scripting thread, a reference to the queue of the second scripting thread for use by the first scripting thread to send messages to the second scripting thread.

11. (Previously Presented) The method of claim 8 further comprising:

creating a message object; inserting a message from the first scripting thread into the message object; and

placing a reference to the message object into the queue of the second scripting thread so that the second scripting thread can access the message.

12. (Previously Presented) The method of claim 11 further comprising:

sending a signal from the first scripting thread to the second scripting thread to indicate to the second scripting thread that a new message has been sent to the second scripting thread.

13. (Previously Presented) The method of claim 11, wherein in response to the message further comprising:

inserting a flag in the message object to indicate that it is being responded to; and

placing a reference to the message object in the queue of the first scripting thread.

14. (Currently Amended) A method for compiling a program having a plurality of sections, the method comprising:

creating, for each section of the program, a scripting thread that executes a script for compiling ~~each~~ the section, wherein the script is independent of the program; and

creating a control thread to asynchronously communicate with each of the scripting threads so that commands can be issued from the control thread to the scripting threads in parallel.

15. (Original) A computer-readable medium having stored thereon computer executable instructions for performing the method of claim 14.

16. (Previously Presented) The method of claim 14 further comprising:

at the control thread, sending updates to a user interface; and

processing commands from the user interface in parallel with asynchronously sending commands to the scripting threads.

17. (Previously Presented) The method of claim 14 further comprising:

creating a queue for the control thread; and

passing, to at least one of the scripting threads, a reference to the control thread's queue for use by the scripting thread to send messages to the control thread.

18. (Currently Amended) A system for compiling a program having a plurality of sections, the system comprising:

a computer;

a script for compiling each section of the program, wherein the script is independent of the program;

a plurality of scripting threads executing on the computer, wherein each section of the program is compiled under the direction of the script executed by a scripting thread of the plurality; and

a control thread executing on the computer for coordinating the activity of the scripting threads by communicating asynchronously with the scripting threads.

19. (Previously Presented) The system of claim 18 further comprising: a means for allowing the control thread to communicate asynchronously with the scripting threads.

20. (Previously Presented) The system of claim 18 further comprising: a plurality of queues, wherein each queue is associated with a scripting thread of the plurality of scripting threads, and wherein each queue is adapted to receive messages from the control thread.

21. (Previously Presented) The system of claim 18 further comprising: a means for sending a signal from the control thread to at least one of the plurality of scripting threads to alert the scripting thread whenever a message is sent to the scripting thread.

22. (Previously Presented) The system of claim 18 further comprising: a script engine executing on the computer, wherein the script engine interprets scripting language commands for each of the plurality of scripting threads and provides a means for sending a signal from the control thread to at least one of the plurality of scripting threads to alert the scripting thread whenever a message is sent to the scripting thread.

23. (Original) The system of claim 18, wherein the computer is a first computer, the system further comprising: at least one second computer in communication with the first computer, wherein at least one of the scripting threads executes on the second computer.

24. (Previously Presented) The system of claim 23 further comprising:

a network link for enabling the first and second computers to communicate with one another;

a means for allowing the scripting thread executing on the second computer to communicate across the network link with the control thread executing on the first computer.

25. (Previously Presented) The system of claim 18 further comprising: a user interface, wherein the control thread is operable to update the user interface without having to wait for the scripting threads to act on messages sent to them by the control thread.

26. (Currently Amended) A system for compiling a program having a plurality of sections, the system comprising:

a server computer;

a script for compiling each section of the program, wherein the script is independent of the program;

a control thread executing on the server computer;

a plurality of client computers, wherein each client computer compiles a section of the plurality of sections, and wherein the client computers are in communication with the server computer; and

a plurality of scripting threads executing on the server computer, wherein each scripting thread directs the compiling activity of a client computer of the plurality of client computers by executing the script, and wherein the control thread sends messages asynchronously to each of the plurality of scripting threads to coordinate their activities.

27. (Original) The system of claim 26, wherein the control thread sends messages asynchronously to each of the plurality of scripting threads to coordinate their activities, thereby resolving interdependencies among different sections of the program that are being compiled.

28. (Previously Presented) The system of claim 26 further comprising:

a one or more control thread queues associated with the control thread; and

a plurality of scripting thread queues, wherein each scripting thread queue is associated with a scripting thread of the plurality of scripting threads, and

wherein the control thread has a reference to each scripting thread queue, and

wherein each scripting thread has a reference to at least one control thread queue that is associated with the scripting thread, thereby enabling the control thread to put one or more of the messages in each scripting thread queue and each scripting thread to put response messages in the associated queue of the control thread.

29. (Previously Presented) The system of claim 26 further comprising:

at least one script stored on the server computer, wherein the script contains instructions for directing the compilation of the program; and

a script engine executing on the server computer to interpret the script, the script engine having an inter-thread signaling mechanism,

wherein the control thread uses signaling mechanism to alert a scripting thread of the plurality of scripting threads whenever the control thread has sent a message to the scripting thread.

30. (Previously Presented) The system of claim 26, wherein the control thread sends messages asynchronously to each of the plurality of scripting threads to coordinate their activities, thereby resolving interdependencies among different sections of the program that are being compiled, the system further comprising:

a plurality of control thread queues associated with the control thread;

a plurality of scripting thread queues, wherein each scripting thread queue is associated with a scripting thread of the plurality of scripting threads, and

wherein the control thread has a reference to each scripting thread queue, and

wherein each scripting thread has a reference to a corresponding control thread queue of the plurality of control thread queues, thereby enabling the control thread to put one or more of the messages in each scripting thread queue and each scripting thread to put response messages in its corresponding control thread queue;

at least one script stored on the server computer, wherein the script contains instructions for directing the compilation of the program; and

a script engine executing on the server computer to interpret the script, the script engine having an inter-thread signaling mechanism, wherein the control thread uses signaling mechanism to alert a scripting thread of the plurality of scripting threads whenever the control thread has sent a message to the scripting thread.